Лабораторна робота №4

Системний реєстр

Вивчити призначення та структуру системного реєстру.

Системний реєстр (Registry) в операційних системах сімейства Windows є централізованою базою даних, де зберігається інформація про конфігурацію операційної системи, додатків та користувацьких налаштувань. Він грає важливу роль у функціонуванні системи, забезпечуючи доступ до налаштувань, параметрів та інших даних, необхідних для роботи системи та програм.

Основні характеристики та структура системного реєстру:

1. **Ієрархічна структура**: Системний реєстр має ієрархічну структуру, схожу на структуру каталогів у файловій системі. Він складається з ключів (keys) та значень (values), які організовані в деревоподібну структуру.
2. **Рівні доступу**: Реєстр має рівні доступу, які контролюють, які процеси можуть зчитувати, записувати або видаляти дані в реєстрі. Це допомагає забезпечити безпеку та недоступність для зловмисників.
3. **Гілки реєстру**: Системний реєстр поділений на кілька гілок, кожна з яких містить певний набір налаштувань та параметрів:
    - **HKEY_CLASSES_ROOT (HKCR)**: Містить інформацію про зареєстровані типи файлів та асоціації з програмами.
    - **HKEY_CURRENT_USER (HKCU)**: Зберігає налаштування, пов'язані з поточним користувачем.
    - **HKEY_LOCAL_MACHINE (HKLM)**: Містить налаштування, які стосуються всієї системи та всіх користувачів.
    - **HKEY_USERS (HKU)**: Містить профілі користувачів, які входять до системи.
    - **HKEY_CURRENT_CONFIG (HKCC)**: Зберігає інформацію про поточну конфігурацію обладнання.
4. **Значення реєстру**: Важлива частина реєстру - це значення, які зберігають конкретні дані. Кожне значення має свій тип даних (наприклад, рядок, число або бітовий масив).
5. **Редагування реєстру**: Для редагування реєстру можна використовувати вбудований редактор реєстру (`regedit`) або інші сторонні програми. Важливо бути обережними при редагуванні реєстру, оскільки неправильні зміни можуть призвести до непрогнозованих наслідків і навіть збоїв системи.

Системний реєстр є однією з найважливіших компонентів операційних систем Windows, і правильне його функціонування необхідно для стабільної та ефективної роботи системи та програм.

Вивчити призначення та методи роботи з утилітою RegEdit.

Утиліта RegEdit (Registry Editor) в операційних системах Windows призначена для перегляду, редагування та керування системним реєстром. Вона надає доступ до всіх ключів і значень, які зберігаються в реєстрі, і дозволяє користувачеві змінювати ці дані відповідно до потреб.

Основні функції та методи роботи з утилітою RegEdit:

1. **Перегляд реєстру**: RegEdit дозволяє користувачеві переглядати структуру реєстру у вигляді дерева, подібно до каталогової структури файлової системи. Користувач може розгортати та згортати гілки, щоб переглядати ключі та їх значення.

2. **Редагування ключів і значень**: Користувач може змінювати значення ключів реєстру або створювати нові ключі та значення. Це дозволяє змінювати налаштування системи, програм та інших параметрів, збережених у реєстрі.
3. **Експорт та імпорт даних реєстру**: Утиліта дозволяє експортувати (зберегти) гілки реєстру у файл .REG, що може бути використаний для резервного копіювання або обміну налаштуваннями з іншими користувачами. Також вона підтримує імпорт (завантаження) даних реєстру з файлів .REG.
4. **Пошук значень реєстру**: RegEdit дозволяє користувачам здійснювати пошук ключів та значень реєстру за певними критеріями. Це допомагає знаходити потрібні параметри у великому реєстрі.
5. **Доступ до спеціальних гілок реєстру**: Утиліта дозволяє користувачам отримати доступ до спеціальних гілок реєстру, таких як HKEY_CLASSES_ROOT, HKEY_CURRENT_USER, HKEY_LOCAL_MACHINE та інші, які містять налаштування для системи, користувачів та програм.

Редагування реєстру може бути потужним інструментом для налаштування системи, але важливо бути обережним при внесенні змін, оскільки неправильні дії можуть призвести до проблем зі стабільністю та функціональністю системи. Перед внесенням будь-яких змін у реєстр рекомендується зробити резервну копію.

Знайти відповідні розділи реєстру в яких є інформація про програми та служби які завантажуються автоматично.

Інформація про програми та служби, які автоматично завантажуються при запуску системи, може бути знайдена в реєстрі Windows. Ось декілька ключів та гілок реєстру, де зазвичай міститься ця інформація:

1. **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**: Цей ключ містить список програм, які автоматично запускаються при вході кожного користувача в систему. Ці програми завантажуються з правами адміністратора.
2. **HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**: Аналогічно попередньому ключу, але цей ключ містить програми, які завантажуються тільки для поточного користувача.
3. **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce**: В цьому ключі містяться програми, які автоматично запускаються лише один раз при наступному запуску системи.
4. **HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce**: Аналогічно попередньому ключу, але для поточного користувача.
5. **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services**: Ця гілка містить інформацію про всі служби, які можуть бути запущені під час завантаження системи. Кожна служба має свій власний ключ з параметрами, включаючи тип завантаження (автоматичний, ручний або вимкнений).
6. **HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows**: У цьому ключі можуть міститися параметри для автоматичного завантаження програм, зокрема параметр "Run", який містить шляхи до програм для автоматичного запуску при вході користувача.
7. **HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run**: Цей ключ призначений для 32-бітних програм на 64-бітних версіях Windows.

Програмно, вивести список усіх програм та служб які завантажуються автоматично для усіх користувачів та поточного користувача.

Лістинг:

```
#include <Windows.h>
#include <iostream>
#include <string>
void PrintStartupPrograms(HKEY hKey) {
    HKEY hKeyRun;
    if (RegOpenKeyEx(hKey, L"Software\\Microsoft\\Windows\\CurrentVersion\\Run", 0,
KEY_READ, &hKeyRun) == ERROR_SUCCESS) {
        DWORD dwIndex = 0;
        WCHAR szValueName[MAX_PATH];
        DWORD dwValueNameLen = MAX_PATH;
        BYTE lpData[MAX_PATH];
        DWORD dwDataLen = MAX_PATH;
        DWORD dwType = 0;

        while (RegEnumValue(hKeyRun, dwIndex, szValueName, &dwValueNameLen, NULL,
&dwType, lpData, &dwDataLen) == ERROR_SUCCESS) {
            if (dwType == REG_SZ || dwType == REG_EXPAND_SZ) {
                std::wstring valueName(szValueName, dwValueNameLen);
                std::wstring valueData(reinterpret_cast<wchar_t*>(lpData), dwDataLen /
sizeof(wchar_t));
                std::wcout << L"Program: " << valueName << L", Path: " << valueData <<
std::endl;
            }
            dwIndex++;
            dwValueNameLen = MAX_PATH;
            dwDataLen = MAX_PATH;
        }
        RegCloseKey(hKeyRun);
    }
    else {
        std::wcerr << L"Failed to open registry key." << std::endl;
    }
}
int main() {
      std::wcout << L"Startup programs for all users:" << std::endl;
      PrintStartupPrograms(HKEY_LOCAL_MACHINE);
      std::wcout << std::endl << L"Startup programs for current user:" <<std::endl;
      PrintStartupPrograms(HKEY_CURRENT_USER);
      return 0;
```

}Пояснення:

Цей код на мові програмування C++ призначений для виведення списку усіх програм та служб, які автоматично завантажуються при запуску системи для всіх користувачів та поточного користувача. Ось пояснення до кожної частини коду:

1. **Включення необхідних заголовочних файлів:** Даний рядок включає заголовочні файли, необхідні для роботи з функціями Windows API та стандартним виведенням в консоль.
2. **Оголошення функцій:**
    - **PrintStartupPrograms:** Функція, яка виводить інформацію про всі програми, які автоматично завантажуються при запуску системи.
    - **main:** Головна функція програми, яка викликає функцію PrintStartupPrograms для всіх користувачів.
3. **Виведення програм для всіх користувачів:**
    - Функція `PrintStartupPrograms` відкриває відповідний розділ реєстру для всіх користувачів за допомогою ключа `HKEY_LOCAL_MACHINE`.
    - Вона викликає `RegOpenKeyEx` для відкриття ключа "Software\Microsoft\Windows\CurrentVersion\Run", де зберігається інформація про програми, що автоматично запускаються.

- Функція використовує `RegEnumValue`, щоб перелічити усі значення в цьому розділі реєстру та вивести інформацію про кожну програму.

4. **Виведення програм для поточного користувача:**
   - Аналогічно, функція `PrintStartupPrograms` відкриває відповідний розділ реєстру для поточного користувача за допомогою ключа `HKEY_CURRENT_USER`.
   - Вона також використовує `RegEnumValue` для виведення інформації про програми, що автоматично запускаються для цього користувача.

Додати програмно до автозавантаження програм для поточного користувача завантаження програми WinWord або іншої.

Лістинг:

```
#include <Windows.h>
#include <iostream>

void AddToStartup(const std::wstring& appName, const std::wstring& appPath) {
    HKEY hKey;
    // Відкриття або створення ключа реєстру для поточного користувача
    if (RegOpenKeyEx(HKEY_CURRENT_USER,
L"Software\\Microsoft\\Windows\\CurrentVersion\\Run", 0, KEY_SET_VALUE, &hKey)
== ERROR_SUCCESS) {
        // Додавання або оновлення значення для ключа реєстру
        if (RegSetValueEx(hKey, appName.c_str(), 0, REG_SZ, (const
BYTE*)appPath.c_str(), (appPath.size() + 1) * sizeof(wchar_t)) == ERROR_SUCCESS)
{
            std::wcout << L"Програма успішно додана до автозавантаження для
поточного користувача." << std::endl;
        } else {
            std::wcerr << L"Помилка додавання програми до автозавантаження." <<
std::endl;
        }
        RegCloseKey(hKey);
    } else {
        std::wcerr << L"Помилка відкриття реєстру для додавання програми до
автозавантаження." << std::endl;
    }
}

int main() {
    std::wstring appName = L"Microsoft Word"; // Ім'я, під яким додається
програма до автозавантаження
    std::wstring appPath = L"C:\\Program Files\\Microsoft
Office\\root\\Office16\\WINWORD.EXE"; // Шлях до виконуваного файлу програми

    AddToStartup(appName, appPath);

    return 0;
}
```

Пояснення:

Функція `AddToStartup`:

```
cpp
```

- `void AddToStartup(const std::wstring& appName, const std::wstring& appPath) {`

Ця функція отримує два параметри: `appName` (ім'я програми) та `appPath` (шлях до виконуваного файлу програми).

- **Відкриття ключа реєстру**:

```
cpp
•  HKEY hKey;
if (RegOpenKeyEx(HKEY_CURRENT_USER,
L"Software\\Microsoft\\Windows\\CurrentVersion\\Run", 0, KEY_SET_VALUE, &hKey)
== ERROR_SUCCESS) {
```

Викликається функція `RegOpenKeyEx`, яка відкриває або створює ключ реєстру для поточного користувача в розділі "Software\Microsoft\Windows\CurrentVersion\Run". Функція повертає значення типу `HKEY`, яке потрібно закрити після використання.

- **Додавання або оновлення значення в реєстрі**:

```
cpp
•  if (RegSetValueEx(hKey, appName.c_str(), 0, REG_SZ, (const
BYTE*)appPath.c_str(), (appPath.size() + 1) * sizeof(wchar_t)) == ERROR_SUCCESS)
{
```

За допомогою функції `RegSetValueEx` додається або оновлюється значення для ключа реєстру. У нашому випадку, значенням є шлях до виконуваного файлу програми. `REG_SZ` вказує, що значення є строкою (звичайним текстом).

- **Закриття ключа реєстру**:

```
cpp
•  RegCloseKey(hKey);
```

Після завершення роботи з реєстром ключ потрібно закрити.

- **Виведення повідомлення про результат**:

```
cpp
•  std::wcout << L"Програма успішно додана до автозавантаження для поточного
користувача." << std::endl;
```

Виводиться повідомлення про успішне додавання програми до списку автозавантаження.

- **Функція `main`**:

```
cpp
int main() {
    std::wstring appName = L"Microsoft Word";
    std::wstring appPath = L"C:\\Program Files\\Microsoft
Office\\root\\Office16\\WINWORD.EXE";

    AddToStartup(appName, appPath);

    return 0;
}
```

У функції `main` визначені інформація про програму та шлях до її виконуваного файлу, після чого викликається функція `AddToStartup` для додавання програми до автозавантаження.

Вивести список повторно, та показали що зареєстрована програма є у списку.

Лістинг:

```
bool IsAppInStartup(const std::wstring& appName) {
    HKEY hKey;
    if (RegOpenKeyEx(HKEY_CURRENT_USER,
L"Software\\Microsoft\\Windows\\CurrentVersion\\Run", 0, KEY_QUERY_VALUE, &hKey)
== ERROR_SUCCESS) {
        DWORD type;
        DWORD dataSize = 0;
        if (RegQueryValueEx(hKey, appName.c_str(), NULL, &type, NULL, &dataSize)
== ERROR_SUCCESS) {
            return true;
        }
        RegCloseKey(hKey);
    }
    return false;
}
int main() {
    std::wstring appName = L"Microsoft Word";
    std::wstring appPath = L"C:\\Program Files\\Microsoft
Office\\root\\Office16\\WINWORD.EXE";

    if (IsAppInStartup(appName)) {
        std::wcout << L"Програма \"" << appName << L"\" вже зареєстрована для
автозавантаження." << std::endl;
    } else {
        AddToStartup(appName, appPath);
    }

    return 0;
}
```

Пояснення:

Функція `IsAppInStartup` призначена для перевірки того, чи є програма з вказаним ім'ям зареєстрована у списку автозавантаження для поточного користувача в реєстрі Windows.

1. **Параметри функції**:
    - `appName`: Рядок, що представляє ім'я програми, яке потрібно перевірити.
2. **Повертає значення**:
    - Логічне значення `true`, якщо програма з вказаним ім'ям зареєстрована у списку автозавантаження.
    - Логічне значення `false`, якщо програма з вказаним ім'ям не зареєстрована у списку автозавантаження або виникла помилка під час перевірки.
3. **Робота функції**:
    - Відкривається ключ реєстру `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run`, де зберігаються дані про автозавантаження.
    - Виконується спроба отримати значення для вказаного імені програми (`appName`) у реєстрі.
    - Якщо значення успішно отримано, це означає, що програма вже зареєстрована у списку автозавантаження, тому функція повертає `true`.
    - Якщо значення не знайдено або виникла помилка під час отримання, функція повертає `false`.

Ця функція дозволяє вам перевірити, чи вже налаштована програма для автозавантаження, щоб уникнути повторного додавання її до списку.

Вивести список усіх завдань, які зареєстровані у планувальнику задач системи. Інформації отримати з відповідного розділу реєстру, як для усіх користувачів так і для поточного користувача.

Лістинг:

```
#include <Windows.h>
#include <iostream>
#include <string>

void PrintTasks(HKEY hKey) {
    DWORD index = 0;
    TCHAR szSubKeyName[256];
    DWORD dwSize = ARRAYSIZE(szSubKeyName);

    while (RegEnumKeyEx(hKey, index++, szSubKeyName, &dwSize, NULL, NULL, NULL,
NULL) == ERROR_SUCCESS) {
        std::wcout << szSubKeyName << std::endl;
        dwSize = ARRAYSIZE(szSubKeyName);
    }
}

int main() {
    // Для всіх користувачів
    HKEY hKeyAllUsers;
    if (RegOpenKeyEx(HKEY_LOCAL_MACHINE, L"SOFTWARE\\Microsoft\\Windows
NT\\CurrentVersion\\Schedule\\TaskCache\\Tasks", 0, KEY_READ, &hKeyAllUsers) ==
ERROR_SUCCESS) {
        std::wcout << L"Tasks for all users:" << std::endl;
        PrintTasks(hKeyAllUsers);
        RegCloseKey(hKeyAllUsers);
    }

    // Для поточного користувача
    HKEY hKeyCurrentUser;
    if (RegOpenKeyEx(HKEY_CURRENT_USER, L"SOFTWARE\\Microsoft\\Windows
NT\\CurrentVersion\\Schedule\\TaskCache\\Tasks", 0, KEY_READ, &hKeyCurrentUser)
== ERROR_SUCCESS) {
        std::wcout << L"Tasks for current user:" << std::endl;
        PrintTasks(hKeyCurrentUser);
        RegCloseKey(hKeyCurrentUser);
    }

    return 0;
}
```

Пояснення:

Цей код на C++ призначений для виведення списку усіх завдань, які зареєстровані у планувальнику задач системи. Він використовує функції роботи з реєстром Windows для доступу до відповідних розділів, де зберігається інформація про ці завдання.

По-перше, програма відкриває розділ реєстру для всіх користувачів та поточного користувача, де зберігається інформація про завдання планувальника задач. Для цього вона використовує функцію `RegOpenKeyEx`. Потім вона викликає функцію `PrintTasks`, яка виводить список усіх підключень у цьому розділі реєстру.

Функція `PrintTasks` використовує `RegEnumKeyEx` для послідовного отримання назв усіх підключень у вказаному розділі реєстру. Потім вона виводить кожне з цих імен на екран.

На виході ви отримуєте список усіх завдань, які зареєстровані у планувальнику задач для всіх користувачів і для поточного користувача.

Зробити програмно копію будь якого розділу реєстру у файл відповідного формату .reg

Лістинг:

```
#include <windows.h>
#include <stdio.h>

// Функція для копіювання розділу реєстру у файл формату .reg
void CopyRegistryKeyToFile(const wchar_t* keyPath, const wchar_t* filePath) {
    // Відкриття розділу реєстру для читання
    HKEY hKey;
    if (RegOpenKeyEx(HKEY_CURRENT_USER, keyPath, 0, KEY_READ, &hKey) !=
ERROR_SUCCESS) {
        printf("Помилка відкриття розділу реєстру.\n");
        return;
    }

    // Відкриття файлу для запису
    FILE* file;
    if (_wfopen_s(&file, filePath, L"w") != 0) {
        printf("Помилка відкриття файлу.\n");
        RegCloseKey(hKey);
        return;
    }

    // Запис у файл заголовку формату .reg
    fwprintf(file, L"Windows Registry Editor Version 5.00\n\n");
    fwprintf(file, L"[%s]\n", keyPath);

    // Зчитування та запис кожного значення реєстру у файл
    wchar_t valueName[255];
    DWORD valueNameSize = sizeof(valueName) / sizeof(valueName[0]);
    DWORD valueType;
    BYTE valueData[4096];
    DWORD valueDataSize = sizeof(valueData);

    for (DWORD i = 0; ; i++) {
        valueDataSize = sizeof(valueData);
        valueNameSize = sizeof(valueName);
        if (RegEnumValue(hKey, i, valueName, &valueNameSize, NULL, &valueType,
valueData, &valueDataSize) != ERROR_SUCCESS) {
            break;
        }

        // Перевірка типу значення
        const wchar_t* valueTypeStr = L"";
        switch (valueType) {
            case REG_SZ:
                valueTypeStr = L"\"";
                break;
            case REG_DWORD:
                valueTypeStr = L"dword:";
                break;
            // Додаткові перевірки можна додати для інших типів значень
        }

        // Запис значення у файл
        fwprintf(file, L"%s%s=%s\n", valueName, valueTypeStr, valueData);
    }
```

```
    // Закриття розділу реєстру та файлу
    RegCloseKey(hKey);
    fclose(file);
}

int main() {
    // Шлях до розділу реєстру та файлу для збереження
    const wchar_t* keyPath =
L"Software\\Microsoft\\Windows\\CurrentVersion\\Run";
    const wchar_t* filePath = L"C:\\Temp\\RegistryBackup.reg";

    // Виклик функції копіювання розділу реєстру у файл
    CopyRegistryKeyToFile(keyPath, filePath);

    printf("Розділ реєстру успішно скопійовано у файл %ls.\n", filePath);
    return 0;
}
```

Пояснення:

Цей код написаний на мові C++ і використовує бібліотеку Windows API для роботи з реєстром Windows.

1. Спочатку оголошується функція `CopyRegistryKeyToFile`, яка приймає два параметри: шлях до розділу реєстру `keyPath` і шлях до файлу `filePath`, в який буде збережено копію розділу реєстру у форматі .reg.
2. У функції відкривається розділ реєстру для читання за допомогою функції `RegOpenKeyEx`. Якщо відкриття розділу реєстру не вдалося, програма виводить повідомлення про помилку і завершує свою роботу.
3. Далі відкривається файл для запису за допомогою функції `fopen_s`. Якщо відкриття файлу не вдалося, програма також виводить повідомлення про помилку і завершує свою роботу.
4. Після відкриття файлу записується заголовок формату .reg за допомогою функції `fwprintf`.
5. Наступним кроком є перебір всіх значень реєстру за допомогою циклу `for`. Для цього використовується функція `RegEnumValue`, яка зчитує кожне значення реєстру в розділі.
6. Значення реєстру додаються до файлу .reg за допомогою функції `fwprintf`. В залежності від типу значення (наприклад, рядок або DWORD), додаються відповідні підказки.
7. Функція завершується закриттям розділу реєстру і файлу за допомогою функцій `RegCloseKey` і `fclose`.
8. У функції `main` встановлюються шляхи до розділу реєстру та файлу для збереження, і викликається функція `CopyRegistryKeyToFile` для копіювання розділу реєстру у файл.

За допомогою текстового редактора створити REG файл, за допомогою якого в реєстр у відповідний розділ буде внесено інформацію про асоціацію відкриття файлів .ttt програмою notepad.

Вміст текстового файлу:

```
[HKEY_CLASSES_ROOT\.ttt]
@="tttfile"

[HKEY_CLASSES_ROOT\tttfile]
@="TTT File"
```

```
"PerceivedType"="text"

[HKEY_CLASSES_ROOT\tttfile\shell]

[HKEY_CLASSES_ROOT\tttfile\shell\open]

[HKEY_CLASSES_ROOT\tttfile\shell\open\command]
@="\"notepad.exe\" \"%1\""
```

Пояснення:

Цей REG файл створює асоціацію між розширенням файлу .ttt і програмою Notepad для їх автоматичного відкриття. Давайте розглянемо кожну частину файлу:

1. `[HKEY_CLASSES_ROOT\.ttt]`: Ця лінія вказує на ключ в реєстрі, який відповідає розширенню .ttt.
2. `@="tttfile"`: Цей запис присвоює значення "tttfile" для розширення .ttt, що означає, що розширення .ttt буде асоційоване з класом tttfile.
3. `[HKEY_CLASSES_ROOT\tttfile]`: Цей ключ створює новий клас файлів з назвою "TTT File".
4. `@="TTT File"`: Цей запис присвоює людсько-читабельну назву "TTT File" для класу файлів tttfile.
5. `"PerceivedType"="text"`: Цей запис вказує, що файл типу "text", що означає текстовий файл.
6. `[HKEY_CLASSES_ROOT\tttfile\shell]`: Цей ключ створює розділ "shell" для класу файлів tttfile, що дозволяє визначити контекстне меню.
7. `[HKEY_CLASSES_ROOT\tttfile\shell\open]`: Цей ключ встановлює команди, пов'язані з дією "відкриття".
8. `[HKEY_CLASSES_ROOT\tttfile\shell\open\command]`: Цей ключ містить команду, яка виконується при відкритті файлу. Тут вказано, що програма Notepad буде відкривати файли .ttt.
9. `@="\"notepad.exe\" \"%1\""`: Цей запис вказує, що для відкриття файлу використовується notepad.exe, де "%1" - це спеціальний параметр, який вказує на шлях до відкриваємого файлу.